IN-81-TM
91506
p. 8

Planning and Scheduling Research at NASA Ames Research Center

PETER FRIEDLAND

AI RESEARCH BRANCH, MAIL STOP 244-17
NASA AMES RESEARCH CENTER
MOFFETT FIELD, CA 94035

(NASA-TM-107838) PLANNING AND SCHEDULING RESEARCH AT NASA AMES RESEARCH CENTER (NASA) 8 p

N92-26790

Unclas

G3/81 0091506

NASA Ames Research Center

Artificial Intelligence Research Branch

Technical Report FIA-90-11-01-01

November, 1990

Planning and Scheduling Research at NASA Ames Research Center

Peter Friedland

Artificial Intelligence Research Branch, Mail Stop 244-17
NASA Ames Research Center
Moffett Field, Ca. 94035

Abstract

Planning and scheduling is the area of artificial intelligence research that focuses on the determination of a series of operations to achieve some set of (possibly) interacting goals and the placement of those operations in a timeline that allows them to be accomplished given available resources. This paper describes work in this area at the NASA Ames Research Center ranging from basic research in constraint-based reasoning and machine learning, to the development of efficient scheduling tools, to the application of such tools to complex Agency problems.

1 Introduction

A major component of human problem-solving is the ability to take a set of goals and determine the actions that must be undertaken to achieve those goals. Of great importance for complex, realworld problems of the type faced by NASA is the need to consider constraints of many types, including resource and time limitations, and the possibility of conflict with other agents in the problem-solving environment. This process is commonly broken into two components: *planning*, which is the analysis of goals and the determination of a sequence of operations likely to achieve those goals; and *scheduling*, which is the assignment of resources to and the temporal placement of those operations.

This paper describes planning and scheduling research within NASA's artificial intelligence research laboratory at the Ames Research Center. It includes the following topics: Constraint-Based Scheduling; Learning and Performance Improvement in Scheduling; Integration of Planning, Scheduling, and Control (The ERE Project); Multi-Agent Planning (The GEMPLAN Project); and Adaptive Planning.

2 Constraint-Based Scheduling

The constraint-based scheduling project, led by Monte Zweben and including Ellen Drascher, Megan Eskey, Todd Stock, and Will Taylor, is concentrating on three major activities: the development of a generic scheduling and rescheduling tool; the study of iterative improvement search algorithms for scheduling; and the application and deployment of the scheduling tool at the Kennedy Space Center (KSC) for Space Shuttle (STS) processing.

The scheduling tool being developed contains an activity description language, an extensible constraint language, a domain description language, a search control rule system, and an interactive scheduling interface. Users declare the activities they wish to schedule with the activity language, relate these activities to each other and to objects in the domain with constraints, and provide domain scheduling knowledge in the form of search control rules. The system then takes this information and finds times and resource assignments for each task such that all the domain constraints are met. One important point to stress is that the system allows one to express constraints on any kind of time-varying information; that is, it is not restricted to modeling only resource availability over time. Examples of this include device states, switch and valve positions, locations, and sensor values. Further, the activity language allows one to express the effects that activities have on these "state variables" in addition to the changes tasks make to resource availability.

In 1990, we will complete the development of the first version of this tool. In 1991, we plan to concentrate on two major topics for tool development. The first is performance; the tool will be optimized for efficiency. The second topic is utility analysis and optimization. In many cases it is essential to develop schedules that *minimize* lateness or work-in-process time. We will extend the search control rule system to support this global optimization criteria.

One of the principal goals of this work is to develop efficient algorithms for rescheduling. In most applications, the *a priori* synthesis of a schedule is important, but equally important is the ability to reactively modify a schedule in reaction to changes that occur during its execution. We have developed rescheduling algorithms that allow users to modify tasks in terms of their start and end times, their constraints, their resource requirements, and their durations.

Our desire for efficient dynamic rescheduling algorithms has resulted in the exploration of iterative improvement scheduling algorithms. These techniques differ from traditional algorithms in that they incrementally repair complete solutions to the scheduling problem rather than systematically extending a partial solution to the problem. Specifically, we have developed a framework called "Constraint-Based Simulated Annealing" which converges to a solution by making local repairs to the violated constraints of some approximately correct schedule. We have developed a number of versions of the Constraint-Based Simulated Annealing algorithm with two major results. First, the new algorithm is at least twice as fast on test problems as conventional scheduling techniques. Second, it is an "anytime" algorithm; that is, at any point the algorithm can be stopped and a solution is returned, with the solution improving the longer the algorithm runs.

In 1991, we plan to continue empirical experiments with the iterative improvement algorithms. In addition, we will begin to tackle the extreme combinatoric nature of large scheduling problems by taking advantage of the inherently parallel nature of the Constraint-Based Simulated Annealing algorithm and attempting an implementation on the massively parallel Connection Machine available at Ames.

The final activity in this project is the deployment of the scheduling tool at KSC for STS processing. Ames, Lockheed AI Center (LAIC), and Lockheed Space Operations Center (LSOC) have teamed to augment the existing planning and scheduling tools available at KSC, with a potential major impact on STS ground operations. We have begun working with KSC schedulers to determine their needs from both a hardware and software perspective. Their current tools are deficient in four major ways. First, they do not support full scheduling and resource allocation. They only provide a scheduling capability comparable to PC project management tools (i.e., PERT/CPM) and have very limited resource leveling capabilities. Second, their tools do not enable the schedulers to represent any constraints other than predecessor and succesor relations beteween tasks. This is a problem because some tasks, such as hazardous tasks, are not causally required to be before or after another task, but instead cannot be accomplished in parallel with other tasks. These tools require schedulers to commit to an arbitrary ordering. Additionally, they cannot represent temporally changing information such as whether or not the shuttle bay doors are open. They implicitly code this by requiring the open door task to precede other tasks. However, if the doors are ever closed for some unscheduled reason, their systems have a difficult time rescheduling. The third weakness of their current approach is that they do not have a interactive graphical interface to their schedule; their process is mainly paper-driven. Finally, they do not have the ability to reactively reschedule; they must start the scheduling from scratch resulting in a completely new schedule that unnecessarily changes much of the schedule. In 1991, we plan to deliver an interactive scheduling tool that is effectively integrated into the Shuttle Processing Data Management System (SPDMS-II). We plan to support this tool for approximately one year, after which we believe it will be officially adopted and supported by the SPDMS-II effort.

3 Learning and Performance Improvement in Scheduling

Learning is perhaps the single most important characteristic of human intelligence. It enables us to improve our abilities over time and to become safer and more robust problem-solvers by avoiding the repetition of errors. Machine learning is the branch of artificial intelligence that focuses on duplicating such behavior within computational systems and is a pervasive theme of artificial intelligence research at Ames.

This project, led by Steve Minton and including Megan Eskey, Andy Philips, and Monte Zweben, is conducting experiments with machine learning as a method for improving scheduling systems. In previous research at Carnegie-Mellon University, it was shown that the performance of a scheduler can be greatly improved if the system recognizes resource bottlenecks and then chooses the resources for an activity requiring a congested resource before choosing activity times. Therefore, we believe that a system should learn when to change its search strategy by analyzing its search progress and learning the general conditions under which a resource bottleneck is likely to occur. We have implemented an analytical learning technique, called Plausible Explantion-Based Learning (PEBL), that accomplishes this.

PEBL extends standard Explantion-Based Learning (EBL) with the addition of an empirical component. EBL is an analytical learning technique in which a system proves that a specific instance is a member of a more general class. We call such a proof the explanation of the instance. The system then derives the weakest conditions under which the explanation holds. If these weakest conditions hold then the object is guaranteed to be a member of the general concept. Systems that perform this process are learning concise and general concept descriptions that are usable later without explanation.

The empirical component is necessary because it is not sufficient to conclude that a chronic resource bottleneck exists from only a single example; multiple examples are needed to gain confidence in the conclusion that a chronic bottleneck exists. When the scheduler reaches a backtracking point, the system tries to explain why it failed. It can do so either by "re-playing" previous explanations or synthesizing new ones. If a previous explanation is used and is successful, then its probability of being accurate is increased. When the confidence in a given explanation reaches some threshold, it is transformed into a search control rule that alters the default search strategy accordingly. New explantions are stored away until their confidences reach the threshold or until they are considered useless. In 1990 we will complete the development of this technique. In 1991, we plan to augment our learning techniques to learn constraint orderings and value preferences. We also plan to extend our learning techniques to the iterative improvement search algorithms.

4 Integration of Planning, Scheduling, and Control (ERE)

The Entropy Reduction Engine (ERE) project, led by Mark Drummond and including John Bresina, Rich Levinson, Andy Philips, and Keith Swanson, is a focus for research on selecting and scheduling actions in a way that takes seriously the likelihood of action execution failure. There are two main subgoals for this research. First, we are doing theoretical and applied work to integrate the representations and methods of AI planning with those of classical scheduling. Our second research subgoal is to make sense of planning and scheduling in terms of modern discrete event control theory.

Traditional AI planning deals with the selection of actions which are relevant to achieving given goals. Various disciplines, principally Operations Research, and more recently AI, have been concerned with the scheduling of actions; that is, with sequencing actions in terms of metric temporal and resource constraints. Most of this work in scheduling remains theoretically and pragmatically disconnected from planning. By integrating action selection and action sequencing we expect to be able to provide a coherent theory of planning and scheduling that can be directly implemented as useful software tools.

Most planning and scheduling work assumes that the job of the system is done when a plan or schedule has been generated. This view is hopelessly optimistic since actions, once selected and sequenced, often fail during subsequent execution. In the ERE project, we view the business of planning and scheduling as that of controlling the behavior of an environment to satisfy certain user-specified goals. A planning or scheduling system cannot simply produce a plan or schedule and then vanish; the system must instead persist in its attempt to drive the environment's behavior in *goal-achieving directions.* Under our view neither planning nor scheduling can be a single-shot effort, but rather a process of participation, where the system must attempt to guide and coerce the environment to conform with given behavioral constraints.

We divide the overall system into three components: reaction, projection, and problem reduction. The reaction component is responsible for producing behavior in the environment, and has a competence independent from the other two components. This independence means that the reaction component does not depend on the existence of a plan to act. The projection component explores possible futures and compiles appropriate reactions. These compiled reactions are expressed as situation-action rules which we call Situated Control Rules (SCRs). When available, these SCRs are given preference by the reaction component during action selection.

Our projection process considers events under the system's control and external events caused by the environment or other agents. Projection uses a domain causal theory represented as a "plan net." Uninformed projection is simply a search through the space of possible event sequences allowed by the plan net and, thus, is infeasible in realistic domains. To achieve efficiency, projection should be controlled. We are considering two ways to control projection: first, by projecting actions selected with reference to the systems overall behavioral constraints; and, second, by limiting the projection of external events with a model of the probability of event occurrence.

Behavioral constraints are expressed in a language based on branching temporal logic. In this language it is possible to express constraints of achievement, maintenance, and prevention. Unfortunately, these constraints are often not in a form which can usefully control the temporal projection search. We are using the third system component, problem reduction, to translate behavioral constraints into search advice for the temporal projection. The problem reduction component, REAPPR, uses domain and problem specific planning expertise to recursively decompose problems into appropriate subproblems. This search through the reduction space can provide guidance to the projection component. In the way that projection informs reaction, so does reduction inform projection.

The eventual goal of the ERE project is a set of software tools for designing and deploying integrated planning and scheduling systems that are able to effectively control their environments. To produce such software tools, we are working towards a better understanding of the theoretical aspects of action selection and sequencing in terms of action execution. Work in this project thus involves both theory and implementation. We are working with others to define a set of benchmark problems and evaluation metrics. With these benchmarks and metrics we will be able to more objectively and comparatively analyze the performance of our architecture. We plan to implement the benchmark problems in software simulations and in a hardware test bed.

Current theoretical work in the group addresses the problem of when to plan and when to act; the integration of problem reduction with temporal projection; probabilistically controlled filtered beam search with anytime properties; and closed-loop hierarchical control systems.

Current implementational work in the group revolves around a set of reactive planning and control problems grounded in a Sun and X11 based simulation called the Reactive Tile World. We are currently extending the existing REAPPR problem reduction system and integrating it with existing temporal projection code.

5 Multi-Agent Planning (GEMPLAN)

This work, led by Amy Lansky and including Andrew Philpot, focuses on the problem of generating multiagent plans for domains with complex coordination requirements. Thus, it deals with both action generation and action ordering, as well as scheduling issues such as resource allocation and timing. Over the past year, the primary activity on this project has been further development and enhancement of the GEMPLAN multiagent planner, which Dr. Lansky originally developed at SRI, International. The system is now a fully domain-independent multiagent planner, with capabilities exceeding those of many state-of-the-art planning systems.

Work on GEMPLAN in 1990 has two major thrusts. The first is exploring the use of "locality" or domain structure to partition domain information as well as the planning space. Representationally, locality can be used to help handle the frame problem, by explicitly limiting the applicability and scope of effect of constraints and events. More importantly, localized reasoning provides a way of alleviating the costly nature of planning (especially multiagent planning) by partitioning the planning search space into smaller, localized search spaces, and thus may facilitate planning in large domains. GEMPLAN's localized search algorithm handles not only hierarchically partitioned domains, but domains with regional overlap as well--a case that is more complex due to the need to maintain consistency between search spaces.

The second focus of this project is devising new methods of constraint satisfaction to enhance GEMPLAN's repertoire of planning capabilities. GEMPLAN users write their domain description in a general-purpose specification language that enables the use of a variety of different kinds of constraints. These domain specifications are then "compiled" into constraint satisfaction code. Several new constraint forms have been added (and previously implemented constraint forms have been extended) to form the following set of GEMPLAN constraints: constraints on patterns of behavior expressed as regular expressions, a variety of temporal and causal constraints among events, a full implementation of the modal truth criterion for multiagent plans (the attainment and maintenance of state-based conditions), and the decomposition of nonatomic events into patterns of subevents. Nonatomic decomposition, in particular, is done in a way that is more general than in standard hierarchical planners -- the nonatomic events are retained within a plan even after they are expanded, enabling reasoning to occur at multiple levels of abstraction within the same context, as is appropriate for each particular constraint. A metric temporal reasoning facility (similar to Dean's time map manager) has also been implemented, but has not yet been integrated into the current GEMPLAN framework.

A recent topic of (as yet only theoretical) interest is the notion of run-time constraint satisfaction; that is, satisfying certain kinds of constraints during plan execution, and in a way that maintains plan correctness. This will be especially useful for achieving priority requirements on resources. Such requirements are important in multiagent domains, which tend to resolve resource contention using a run-time priority policy. This subject is also related to the notion of disjunctive plans (plans that have multiple possible execution paths) and the resolution of some forms of disjunction at run-time.

GEMPLAN has been applied to several example problems, including multiagent blocks world planning and the Tower of Hanoi. The latter problem is optimally solved with no clues about solving subproblems. (We have both single-agent and multiple agent solutions for this problem, but the single-agent case is definitely the most difficult case for this particular problem). We have also applied GEMPLAN to a small construction domain example (forty-nine actions are generated). This domain includes multiple walls and contractors, and thus requires both resource allocation and coordination of actions occurring within shared regions. This was a useful test of our new localized search code, which can handle regional overlap. We have performed various empirical tests experimenting with different levels of localization and regional overlap in this domain, and results have shown that locality provides great efficiency benefits.

For 1991 and beyond we plan to focus on two primary goals. The first is the application of the current GEMPLAN system (in Quintus Prolog) to a NASA domain. One such domain under active investigation is the planning that goes into very large scale data analysis tasks such as those to be faced by the Earth Observing System series of satellites. The second goal is the development of a new GEMPLAN system in Lisp that investigates several new areas: the integration of preplanning and prescheduling with dynamic run-time planning and scheduling mechanisms; parallel search of independent localities; and associative attachment and tracking of constraints with planned events (the current GEMPLAN system does include a limited capability of this kind).

6 Adaptive Planning

The long-term objective of this research, led by Smadar Kedar and including Lisa Dent, is to augment reactive systems (systems that react to dynamic environments) with the ability to refine their interaction with the world through learning from experience. In particular, we are examining situations in which the system may fail to react appropriately, and would learn from its mistakes.

For future NASA missions, the ability to autonomously react and refine reactions through experience will be needed when human teleoperation of a robot may not be possible. Such scenarios may include teleoperation when the time delay is too long (e.g. unloading payloads from a descent vehicle on Mars), or when low-level actions are difficult for humans to control (due to vibrations or unpredictable movements). In such scenarios, teleoperation commands may need to be more high-level, while fine-grained actions would be generated and refined more autonomously.

The research motivations for this work are two-fold. First, most current reactive systems can only react to situations which have been completely specified *a priori*. For unanticipated conditions, these systems may fail to react at all. More robust reactive systems need to be augmented with capabilities for detecting such failures, repairing them, and augmenting their reactive rules in a general way so as to learn to avoid such failures in the future. Second, most symbolic machine learning approaches to learning from failure are limited in that they do not work in reactive situations, but assume a "plan-then-execute" model of action. These approaches need to be applied and tested in reactive systems.

In 1990, we have focused our initial efforts on problems of failing to react in a calibration and tracking scenario for a robotic hand-eye system. (To calibrate vision with arm movement, for any point in the robot arm coordinate frame, a corresponding point in the vision coordinate frame is found. Once they are calibrated, the robot arm tracks an object across the visual field, and poses above it.) Given an initial set of reactive rules for calibration and tracking, certain exceptional conditions (such as additional objects other than the arm in the visual field during calibration, or a loose gripper) are not accounted for.

To address this problem, we propose a novel reactive system architecture which, along with its reactive rules, has a list of possible error conditions for the arm, vision, and reasoning systems (e.g. lose gripper). As failures happen, a strategy to recover from failure is invoked. In parallel, the cause of the failure is explained using the theory of possible errors (or through an explanation from the human), and then generalized applying explanation-based learning from failure. The generalized conditions of the failure are "compiled-as-needed" into the original set of reactive rules if it is believed that the failure will recur. As an alternative to the automatic modification of the rules, the system can be used as a tool for a user to experiment and refine reactive rules based on suggestions from the system.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE Dates attached	3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE

Titles/Authors - Attached

5. FUNDING NUMBERS

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Code FIA - Artificial Intelligence Research Branch
Information Sciences Division

8. PERFORMING ORGANIZATION REPORT NUMBER

Attached

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Nasa/Ames Research Center
Moffett Field, CA. 94035-1000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Available for Public Distribution

Peter Friedland 5/14/92 BRANCH CHIEF

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Abstracts ATTACHED

14. SUBJECT TERMS	15. NUMBER OF PAGES
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT